



# Tecnológico de Monterrey

## **Actividad en equipo: MongoDB en tu proyecto**

Manuel Eduardo Covarrubias Rodríguez A01737781

Javier Cuatepotzo Hernandez A01737789

Anhuar Maldonado Aguilar A01737767

Juan Daniel Salmerón Mora A01737637

21/05/25

## Contextualización del proyecto

1. En tu proyecto, ¿qué tipo de datos vas a manejar?

Actualmente manejamos principalmente estos datos:

- **Información personal y de contacto** de participantes, mentoras, colaboradoras, coordinadoras y padres/tutores.
  - **Datos académicos** ,idioma, nivel, escolaridad, escuela, grupo.
  - **Datos organizacionales** asociados a sedes y convocatorias ,fechas, estado, coordinadoras asociadas.
2. ¿Por qué sería útil usar una base de datos NoSQL en este contexto?

La flexibilidad de esquema te permite añadir o quitar campos (por ejemplo, historial de notas, fechas de sesión, roles de colaboradores) sin alterar tablas ni ejecutar migraciones. Así puedes adaptar tu modelo de datos al vuelo, acelerar despliegues y evitar ventanas de mantenimiento.

## Diseño del modelo NoSQL

1. Elabora un esquema de al menos una colección principal.

```
// Ejemplo de un documento en db.participantes
{
  _id: ObjectId,
  nombre: "María",
  apellido_paterno: "García",
  apellido_materno: "López",
  edad: 15,
  correo: "maria.garcia@example.com",
  escuela: "Preparatoria ABC",
  escolaridad: "Secundaria",
  permiso_padre_tutor: "firma123.pdf",
}
```

```
idioma: "Español",
estado: "Activo",

padre_o_tutor: {                                // embed del tutor
  nombre: "Juan",
  apellido_paterno: "García",
  apellido_materno: "Martínez",
  correo: "juan.garcia@example.com",
  telefono: "555-1234"
},

grupo: {                                        // embed del grupo asignado
  id_grupo: 10,
  idioma: "Inglés",
  nivel: "Intermedio",
  cupo: 20
},

sede: {                                        // embed de la sede correspondiente
  id_sede: 3,
  nombre: "Sede Centro",
  convocatoria: "Primavera 2025",
  fecha_inicio: ISODate("2025-06-01"),
  estado: "Abierta"
}
}
```

2. Describe los campos de los documentos (nombre, tipo, relaciones internas).

<b>Campo</b>	<b>Tipo</b>	<b>Descripción</b>
<code>_id</code>	ObjectId	Identificador único generado por MongoDB
<code>nombre,</code> <code>apellido_*</code>	String	Datos personales
<code>id_participante</code>	Number	Clave de identificación usada en sql (agregada para mantener la lógica de sql)
<code>edad, idioma</code>	Number / String	Edad y preferencias lingüísticas
<code>correo</code>	String	Correo de contacto
<code>escuela,</code> <code>escolaridad</code>	String	Información académica
<code>permiso_padre</code> <code>_tutor</code>	String	Ruta o nombre de archivo de permiso firmado

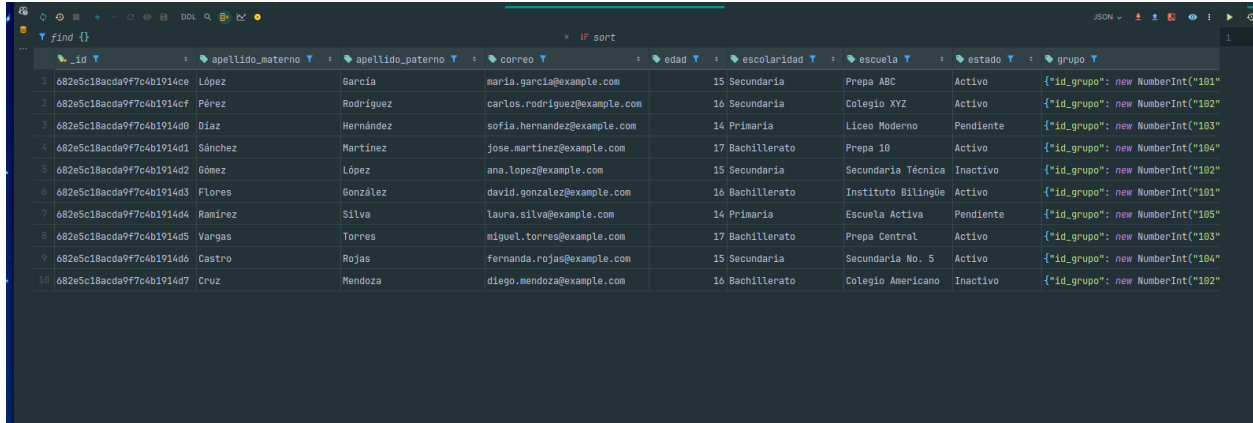
estado	String	“Activo”, “Inactivo”, etc.
padre_o_tutor	Object (embed)	Subdocumento con datos del tutor (evita FK y join)
grupo	Object (embed)	Subdocumento con datos del grupo (idioma, nivel, cupo)
sede	Object (embed)	Subdocumento con datos de la sede (convocatoria, fecha...)

## Documentos en MongoDB:

```

1 {
2   "_id": {"$oid": "602e5c18acd997c4b1914ca"},
3   "apellido-materno": "López",
4   "apellido-paterno": "García",
5   "correo": "maria.garcia@example.com",
6   "sede": 15,
7   "escolaridad": "Secundaria",
8   "escuela": "Prepa ABC",
9   "estado": "Activo",
10  "grupo": {
11    "id_grupo": 101,
12    "idioma": "Inglés",
13    "nivel": "Básico",
14    "cupo": 1
15  },
16  "id_participante": 1,
17  "idioma": "Español",
18  "nombre": "María",
19  "padre_o_tutor": {
20    "id_padre_o_tutor": 10,
21    "nombre": "Juan",
22    "apellido-paterno": "García",
23    "apellido-materno": "Mtz",
24    "correo": "juan.garcia@example.com",
25    "telefono": "555-1001"
26  },
27  "permiso_padre_tutor": "permiso1.pdf",
28  "sede": {
29    "id_sede": 201,
30    "nombre": "Sede Norte",
31    "convocatoria": "Verano",
32    "fecha_inicio": {"$date": "2025-07-01T00:00:00.000Z"},
33    "estado": "Abierta"
34  }
35 },
36 {
37   ...
38 }

```



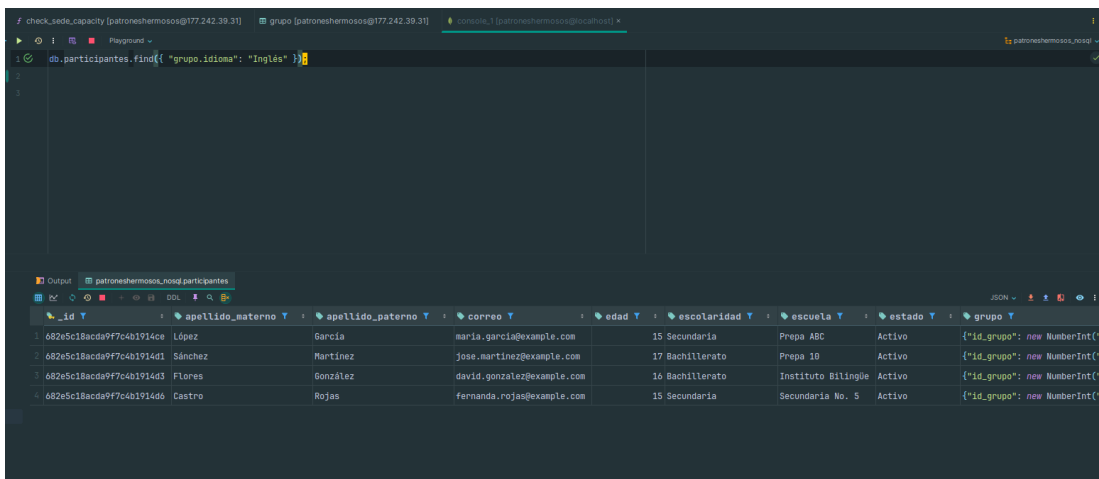
_id	apellido_materno	apellido_paterno	correo	edad	escolaridad	escuela	estado	grupo
682e5c18acda9f7c4b1914ce	López	García	maria.garcia@example.com	15	Secundaria	Prepa ABC	Activo	{"id_grupo": new NumberInt("181")
682e5c18acda9f7c4b1914cf	Pérez	Rodríguez	carlos.rodriguez@example.com	16	Secundaria	Colegio XYZ	Activo	{"id_grupo": new NumberInt("182")
682e5c18acda9f7c4b1914d0	Díaz	Hernández	sofia.hernandez@example.com	14	Primaria	Liceo Moderno	Pendiente	{"id_grupo": new NumberInt("183")
682e5c18acda9f7c4b1914d1	Sánchez	Martínez	jose.martinez@example.com	17	Bachillerato	Prepa 10	Activo	{"id_grupo": new NumberInt("184")
682e5c18acda9f7c4b1914d2	Gómez	López	ana.lopez@example.com	15	Secundaria	Secundaria Técnica	Inactivo	{"id_grupo": new NumberInt("182")
682e5c18acda9f7c4b1914d3	Flores	González	david.gonzalez@example.com	16	Bachillerato	Instituto Bilingüe	Activo	{"id_grupo": new NumberInt("181")
682e5c18acda9f7c4b1914d4	Ramírez	Silva	laura.silva@example.com	14	Primaria	Escuela Activa	Pendiente	{"id_grupo": new NumberInt("185")
682e5c18acda9f7c4b1914d5	Vargas	Torres	miguel.torres@example.com	17	Bachillerato	Prepa Central	Activo	{"id_grupo": new NumberInt("183")
682e5c18acda9f7c4b1914d6	Castro	Rojas	fernanda.rojas@example.com	15	Secundaria	Secundaria No. 5	Activo	{"id_grupo": new NumberInt("184")
682e5c18acda9f7c4b1914d7	Cruz	Mendoza	diego.mendoza@example.com	16	Bachillerato	Colegio Americano	Inactivo	{"id_grupo": new NumberInt("182")

### 3. Justifica el uso de una base de datos documental en lugar de relacional

- Evita operaciones de join en consultas muy frecuentes(e.g. listar participantes con su tutor y sede).
- Lecturas atómicas: al leer un participante obtenga toda su info en un solo fetch.
- Evolución sencilla: si más adelante añado un campo (por ejemplo, país,estado,etc.), no necesito alterar tablas ni migrar datos.

## Implementación práctica

1. Usa MongoDB Shell, MongoDB Compass o MongoDB Atlas.
2. Inserta al menos 10 documentos en tu colección (pueden ser ejemplos representativos).
3. Realiza al menos 3 consultas:
  1. Una consulta con filtro.



```
db.participantes.find({'grupo.idioma': 'Inglés'})
```

_id	apellido_materno	apellido_paterno	correo	edad	escolaridad	escuela	estado	grupo
682e5c18acda9f7c4b1914ce	López	García	maria.garcia@example.com	15	Secundaria	Prepa ABC	Activo	{"id_grupo": new NumberInt("181")
682e5c18acda9f7c4b1914d1	Sánchez	Martínez	jose.martinez@example.com	17	Bachillerato	Prepa 10	Activo	{"id_grupo": new NumberInt("184")
682e5c18acda9f7c4b1914d3	Flores	González	david.gonzalez@example.com	16	Bachillerato	Instituto Bilingüe	Activo	{"id_grupo": new NumberInt("181")
682e5c18acda9f7c4b1914d6	Castro	Rojas	fernanda.rojas@example.com	15	Secundaria	Secundaria No. 5	Activo	{"id_grupo": new NumberInt("184")

## 2. Una actualización o eliminación.

```
db.participantes.find(id_participante: 3));

db.participantes.updateOne(
  { id_participante: 3 },
  { $set: { estado: "Activo" } }
);
```

Output: patroneshermosos\_nosql.participantes

_id	apellido_materno	apellido_paterno	correo	edad	escolaridad	escuela	estado	grupo
14d0	Díaz	Hernández	sofia.hernandez@example.com	14	Primaria	Liceo Moderno	Activo	{"id_grupo": new NumberInt("103"), "idioma": "Ale

## 3. Una consulta con ordenamiento.

```
db.participantes.find().sort({ apellido_paterno: 1 });
```

Output: patroneshermosos\_nosql.participantes

_id	apellido_materno	apellido_paterno	correo	edad	escolaridad	escuela	estado	grupo	
1	682e5c18acda9f7c4b1914ce	López	García	maria.garcia@example.com	15	Secundaria	Prepa ABC	Activo	{"id_grupo": new NumberInt
2	682e5c18acda9f7c4b1914d3	Flores	González	david.gonzalez@example.com	16	Bachillerato	Instituto Bilingüe	Activo	{"id_grupo": new NumberInt
3	682e5c18acda9f7c4b1914d0	Díaz	Hernández	sofia.hernandez@example.com	14	Primaria	Liceo Moderno	Activo	{"id_grupo": new NumberInt
4	682e5c18acda9f7c4b1914d2	Gómez	López	ana.lopez@example.com	15	Secundaria	Secundaria Técnica	Inactivo	{"id_grupo": new NumberInt
5	682e5c18acda9f7c4b1914d1	Sánchez	Martínez	jose.martinez@example.com	17	Bachillerato	Prepa 10	Activo	{"id_grupo": new NumberInt
6	682e5c18acda9f7c4b1914d7	Cruz	Mendoza	diego.mendoza@example.com	16	Bachillerato	Colegio Americano	Inactivo	{"id_grupo": new NumberInt
7	682e5c18acda9f7c4b1914cf	Pérez	Rodríguez	carlos.rodriguez@example.com	16	Secundaria	Colegio XYZ	Activo	{"id_grupo": new NumberInt
8	682e5c18acda9f7c4b1914d6	Castro	Rojas	fernanda.rojas@example.com	15	Secundaria	Secundaria No. 5	Activo	{"id_grupo": new NumberInt
9	682e5c18acda9f7c4b1914d4	Ramírez	Silva	laura.silva@example.com	14	Primaria	Escuela Activa	Pendiente	{"id_grupo": new NumberInt
10	682e5c18acda9f7c4b1914d5	Vargas	Torres	miguel.torres@example.com	17	Bachillerato	Prepa Central	Activo	{"id_grupo": new NumberInt

## Uso de IA generativa

1. En tu entrega, incluye ejemplos de cómo usaste ChatGPT u otra IA:
  1. ¿Qué preguntas hiciste?
    - i. “¿Cómo representar en MongoDB la relación entre participante y sede, optimizando lecturas?”
    - ii. “¿Cuál es la mejor práctica para embed o referenciar documentos en colecciones de MongoDB?”
    - iii. “¿Cómo usar `insertMany` y `updateOne` en el shell de MongoDB?”
  2. ¿Qué código o explicaciones te ayudaron?
    - i. La recomendación de embebido (embed) para datos raramente cambiantes (sede, tutor) fue aceptada tal cual y aplicada en mi esquema.
    - ii. El ejemplo de sintaxis de `insertMany` me sirvió directamente, solo tuve que adaptar los nombres de campos de mi SQL dump.
  3. Evalúa la utilidad de las respuestas (¿te sirvieron tal cual? ¿tuviste que adaptarlas?).
    - i. **Utilidad:** las respuestas fueron claras y con ejemplos.
    - ii. **Adaptaciones:** tuve que ajustar algunos ejemplos porque no seguían la lógica necesaria de patrones hermosos.

## Reflexión final

1. ¿Qué aprendiste al usar MongoDB en tu propio contexto?

Comprendimos cómo la práctica de embed en documentos para parámetros que no cambian tan seguido y aprendimos la increíble flexibilidad que puede llegar a tener MongoDB.



2. ¿Qué ventajas o limitaciones encontraste?

Una ventaja de NoSQL es su esquema flexible (schemaless): Es muy útil para adaptarnos rápidamente a cambios sin necesidad de reestructurar toda la base de datos.

Mongo es más fácil de escalar horizontalmente comparado con una base de datos relacional convencional

Dentro de las limitaciones encontramos que es fácil confundir cuando embebido y cuando referenciar, ahora al embebir podemos cometer el error de duplicar datos, otra limitante es que si ocupamos embebidos en vez referenciar, cuando se quiera actualizar un campo, tenemos que modificarlo en todos los documentos.

3. ¿Cómo te ayudó la IA y cómo podrías usarla mejor en el futuro?

La IA fue una herramienta clave para entender mejor el modelo de datos NoSQL y su aplicación en contextos reales. A futuro, podría optimizar su uso elaborando consultas más precisas y pidiendo ejemplos más detallados que fortalezcan nuestro aprendizaje y capacidades técnicas.

Capturas de uso de IA:

ES correcto tener esto `_id` e `id_participante`:

`_id` ObjectId Identificador único generado por MongoDB  
`id_participante` Number Clave primaria de la tabla relacional?

Thought for a couple of seconds >

Sí, es completamente válido y a menudo útil:

- `_id : ObjectId`
  - Es el identificador interno obligatorio que MongoDB crea para cada documento.
  - Garantiza unicidad y viene optimizado para índices y sharding.
- `id_participante : Number`
  - Es tu clave de negocio (p. ej. la PK que ya usabas en SQL).
  - Te permite conservar referencias legadas, integrarte con otros sistemas o exponer un ID numérico legible.

**Recomendaciones**

1. **Mantén ambos** si necesitas compatibilidad con sistemas existentes o quieres exponer un ID simple.
2. **Índexalo**: crea un índice único sobre `id_participante` para asegurar integridad:

```
js
db.participantes.createIndex({ id_participante: 1 }, { unique: true });
```

3. **Alternativa**: si no requieres la PK numérica, podrías usar solo `_id` (asignándolo manualmente si prefieres números) y eliminar `id_participante`.

En resumen, usar `_id` para MongoDB y `id_participante` como tu identificador de dominio es una práctica

Ask anything

+ Tools

ChatGPT can make mistakes. Check important info.

so if i have participante 1 and its in gruop 201 and I want to add participante 2 to group 201, I need to duplicate the fields in group 201 in the body of participante 2?

Thought for 4 seconds >

If you model `grupo` as a fully embedded sub-document, then **yes**—every participante document that belongs to el mismo grupo 201 tendría su propia copia de esos campos:

```
js                                                                    Copy Edit

// participante 1
{
  ...,
  grupo: { id_grupo: 201, idioma: "Inglés", nivel: "Básico", cupo: 20 }
}

// participante 2
{
  ...,
  grupo: { id_grupo: 201, idioma: "Inglés", nivel: "Básico", cupo: 20 }
}
```

### ¿Es esto un problema?

- **Duplicación de datos:** si mañana cambias `cupo` a 25, tendrías que actualizar **todos** los documentos que embeden ese grupo.
- **Espacio:** cada documento almacena la info completa, pero en la práctica el sobrecoste suele ser pequeño para objetos pequeños.



## 2. Modelo híbrido (embed + referencia)

- Guardas un pequeño "snapshot" de `grupo` (por ejemplo nombre y nivel)
- Y referencias `grupo_id` para datos más voluminosos o cambiantes

```
js Copy Edit
{
  ...,
  grupo_id: 201,
  grupo: { idioma: "Inglés", nivel: "Básico" }
}
```

### ¿Cuándo embeder vs. referenciar?

- **Embed sí:**
  - El sub-documento es pequeño y **rara vez cambia**.
  - Quieres lecturas atómicas y de baja latencia (sin joins).
- **Referenciar sí:**
  - El sub-documento puede cambiar con frecuencia (por ejemplo, cupo o coordinador).
  - Prefieres mantener un solo origen de verdad para la entidad "grupo".

En tu caso, si los grupos cambian muy poco, la duplicación es un coste aceptable y te ahorras joins. Si vas a actualizar mucho la info de los grupos, mejor usar referencia.